(19) **Bundesrepublik Deutschland**
**Deutsches Patent- und Markenamt**

(10) **DE 601 05 063 T2** 2005.08.18

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 352 317 B1**
(21) Deutsches Aktenzeichen: **601 05 063.0**
(86) PCT-Aktenzeichen: **PCT/US01/50119**
(96) Europäisches Aktenzeichen: **01 991 532.1**
(87) PCT-Veröffentlichungs-Nr.: **WO 02/037268**
(86) PCT-Anmeldetag: **19.10.2001**
(87) Veröffentlichungstag der PCT-Anmeldung: **10.05.2002**
(97) Erstveröffentlichung durch das EPA: **15.10.2003**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **18.08.2004**
(47) Veröffentlichungstag im Patentblatt: **18.08.2005**

(51) Int Cl.[7]: **G06F 9/00**

(30) Unionspriorität:
**702224** **31.10.2000** **US**

(84) Benannte Vertragsstaaten:
**AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LI, LU, MC, NL, PT, SE, TR**

(73) Patentinhaber:
**Unisys Corp., Blue Bell, Pa., US**

(74) Vertreter:
**Schieber und Kollegen, 80469 München**

(72) Erfinder:
**SCHOLZ, Wilmer, Karl, Strafford, US; IRWIN, S., James, Stevens, US; TAMRI, Samir, Frazer, US**

(54) Bezeichnung: **ENTWICKLUNGSWERKZEUG FÜR EINEN DIALOGFLUSSINTERPRETER**

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patent- und Markenamt inhaltlich nicht geprüft.

## Beschreibung

### GEBIET DER ERFINDUNG

**[0001]** Die vorliegende Erfindung betrifft allgemein sprachaktivierte interaktive Sprachreaktionssysteme (IVR) und auf ähnliche Systeme, die einen Dialog zwischen einem Menschen und einem Computer einbeziehen. Im Besonderen stellt die vorliegende Erfindung ein Entwicklungswerkzeug für einen Dialogflussinterpreter zum Implementieren von maschinennahen Dialogdetails als auch von Übersetzerobjektklassen zum Handhaben von spezifischen Datentypen (z. B. Währung, Datumsangaben, Stringvariablen, usw.) bereit.

### HINTERGRUND DER ERFINDUNG

**[0002]** Computer sind in unserem Alltag allgegenwärtig geworden. Heute verrichten Computer viel mehr als nur zu rechnen: Supermarktscanner berechnen unsere Lebensmittelrechnung während sie das Ladeninventar verfolgen; computerisierte Fernsprechvermittlungsstellen lenken Millionen von Verbindungen; Bankautomaten (ATMs) gestatten es Menschen, Banktransaktionen von nahezu überall zu führen – die Liste lässt sich weiter fortsetzen. Für die meisten Menschen ist es schwer, sich einen einzigen Tag vorzustellen, an dem sie nicht in irgendeiner Weise mit einem Computer interagieren.

**[0003]** Früher waren Computerbenutzer gezwungen, mit Computern nach den Bedingungen des Computers zu interagieren – über die Tastatur oder die Maus oder seit Neuerem über die Tastaturwahl auf einem Telefon (als DTMF bezeichnet – für Zweiton-Mehrfrequenz). Der Trend bewegt sich jedoch mehr und mehr dahin, Interaktionen zwischen Computern einfacher und benutzerfreundlicher zu gestalten. Ein Weg, um Interaktionen zwischen Computern und Menschen freundlicher zu gestalten, besteht darin, es Menschen und Computern zu gestatten, über gesprochene Worte zu interagieren.

**[0004]** Um einen Dialog zwischen einem Menschen und einem Computer zu ermöglichen, benötigt der Computer zuerst eine Spracherkennungsfähigkeit, um die gesprochenen Worte zu erfassen und sie in irgendeine Form von Computer-lesbaren Daten umzuwandeln, wie z. B. in einen einfachen Text. Als Nächstes benötigt der Computer irgendeinen Weg, um die Computer-lesbaren-Daten zu analysieren und zu bestimmen, was jene Worte in ihrer Verwendung bedeuteten. Eine höhere sprach-aktivierte, stimm-aktivierte oder natürliche Sprache verstehende Anwendung arbeitet typischerweise, indem sie einen schrittweise gesprochenen Dialog zwischen dem Benutzer und dem Computersystem führt, auf dem die Anwendung läuft. Beim Verwenden von herkömmlichen Verfahren legt der Entwickler von derartigen höheren Anwendungen den Quellcode fest, der jeden möglichen Dialog und jeden Schritt von jedem Dialog implementiert. Um eine robuste Anwendung zu implementieren, sieht der Entwickler voraus und handhabt in der Software auf jede mögliche Eingabeaufforderung jede mögliche Benutzerantwort, egal ob derartige Antworten erwartet oder unerwartet sind. Die Last auf dem Entwickler für die höhere Anwendung ist beträchtlich, um derartige maschinennahe Details zu handhaben.

**[0005]** Da die Nachfrage nach sprach-aktivierten Anwendungen gestiegen ist, ist somit die Nachfrage nach Entwicklungressourcen gestiegen. Gegenwärtig übersteigt die Nachfrage nach sprach-aktivierten Anwendungen die zur Codierung der Anwendungen erhältlichen Entwicklungsressourcen. Auch die Nachfrage nach Entwicklern mit der notwendigen Sachkenntnis, um die Anwendungen zu schreiben, übersteigt die Kapazität der Entwickler, die über diese Sachkenntnis verfügen. Folglich besteht eine Notwendigkeit, den Entwicklungsprozess von interaktiven Sprachanwendungen zu vereinfachen und voranzutreiben.

**[0006]** Zusätzlich zu der Zeitdauer, die benötigt wird, um sprachaktivierte Anwendungen zu entwickeln und zu der Fertigkeitsstufe, die erforderlich ist, um diese Systeme zu entwickeln, besteht ein weiterer Nachteil der gegenwärtigen Form von sprach-aktivierter Anwendungsentwicklung darin, dass sie verkäuferspezifisch ist, was einer Wiederverwendung des Codes deutlich entgegenwirkt, wenn der Verkäufer wechselt, und anwendungsspezifisch ist, was bedeutet, dass ein bereits geschriebener Code für eine andere Anwendung nicht wiederverwendet werden kann. Somit besteht auch eine Notwendigkeit, in der Lage zu sein, ein verkäufer-unabhängiges System und einen wiederverwendbaren Code hervorzubringen.

**[0007]** Ein zusätzlicher Hintergrund zu IVR-Systemen kann im US-Patent 6.094.635 vom 25. Juli 2000 mit dem Titel "System and Method for Speech Enabled Application (System und Verfahren für eine sprach-aktivierte Anwendung)"; im US-Patent 5.995.918 vom 30. November 1999 "System and Method for Creating a Language Grammar using a Spreadsheet or Table Interface (System und Verfahren zum Erschaffen einer Programmiersprachengrammatik, die eine Tabellenkalkulations- oder Tabellenschnittstelle verwendet)" und im US-Patent 6.510.411 vom 21. Januar 2003 "Task Oriented Dialog Model, and Manager (Aufgabenorientiertes Dialogmodell und Verwalter)" gefunden werden. Diese Patente sind gemeinsam an die Unisys Corporation übertragen worden.

## ZUSAMMENFASSUNG DER ERFINDUNG

**[0008]** Die vorliegende Erfindung wird in den angefügten Ansprüchen definiert und bezieht sich auf, ist aber nicht notwendigerweise beschränkt auf, Computersoftwareerzeugnisse, die dazu verwendet werden, um Anwendungen zum Ermöglichen eines Dialogs zwischen einem Menschen und einem Computer hervorzubringen. Eine derartige Anwendung könnte in irgendeiner Branche (die Verwendung im Bankwesen, Maklerwesen oder im Internet, usw. einschließend) verwendet werden, wodurch ein Benutzer einen Dialog mit einem Computer unter Verwendung z. B. eines Telefons, Funktelefons oder Mikrofons führt.

**[0009]** Die vorliegende Erfindung stellt die zuvor genannten Notwendigkeiten zufrieden, indem sie ein Entwicklungswerkzeug bereitstellt, das Softwareentwickler von zeitintensiven, technisch herausfordernden Entwicklungsaufgaben löst, indem es dem Entwickler gestattet, verallgemeinerte Befehle an das Entwicklungswerkzeug für einen Dialogflussinterpreter (DFI) oder das DFI-Werkzeug festzulegen. Eine Anwendung instantiiert ein Objekt (d. h. das DFI-Objekt), wobei das Objekt dann Funktionen aufruft, um die Sprachanwendung zu implementieren. Das DFI-Werkzeug besetzt automatisch eine Bibliothek mit Dialogobjekten, die für andere Anwendungen verfügbar sind.

**[0010]** Die Sprachanwendungen, die durch das DFI-Werkzeug hervorgebracht wurden, können als COM-Objekte (Komponentenobjektmodell) implementiert werden, und somit können die Anwendungen einfach in eine Vielzahl von unterschiedlichen Plattformen integriert werden. Eine Anzahl von unterschiedlichen Spracherkennungsmaschinen kann auch unterstützt werden. Die spezielle Spracherkennungsmaschine, die in einer speziellen Anwendung verwendet wird, kann einfach gewechselt werden.

**[0011]** Ein weiterer Aspekt der vorliegenden Erfindung ist die Bereitstellung von "Übersetzer"-Objektklassen, die entworfen wurden, um spezifische Datentypen wie z. B. Währung, numerische Daten, Datumsangaben, Zeitangaben, Stringvariablen, usw. zu handhaben. Diese Übersetzerobjektklassen können einen Nutzen entweder als ein Teil der DFI-Bibliothek von Objekten haben, die oben zum Implementieren von Dialogen beschrieben wurden, oder als eine Unter-Bibliothek, die von einer Dialogimplementierung getrennt ist.

**[0012]** Andere Aspekte der vorliegenden Erfindung werden unten beschrieben.

## KURZE BESCHREIBUNG DER ZEICHNUNGEN

**[0013]** Fig. 1 stellt ein herkömmliches IVR-System schematisch dar.

**[0014]** Fig. 2 ist ein Flussdiagramm eines Verfahrens gemäß der vorliegenden Erfindung für die Entwicklung einer Sprachanwendung.

**[0015]** Fig. 3 ist ein Flussdiagramm, das eine Sprachanwendung aus dem Stand der Technik darstellt.

**[0016]** Fig. 4 ist ein Flussdiagramm eines Verfahrens gemäß der vorliegenden Erfindung für die Entwicklung eines Entwurfs und die Erzeugung einer Datendatei für eine Sprachanwendung.

**[0017]** Fig. 5 ist ein Flussdiagramm eines Verfahrens gemäß der vorliegenden Erfindung für die Erzeugung einer Sprachanwendung.

**[0018]** Fig. 6(a) und Fig. 6(b) liefern einen Vergleich der Menge an Code, die von einem Entwickler geschrieben wurde, der ein System aus dem Stand der Technik verwendet, mit derjenigen, die von einem Entwickler geschrieben wurde, der ein System gemäß der vorliegenden Erfindung verwendet.

**[0019]** Fig. 7 ist ein schematisches Diagramm, das Funktionen und gemeinsam benutzte Objekte gemäß der vorliegenden Erfindung darstellt.

## DETAILLIERTE BESCHREIBUNG DER BEVORZUGTEN AUSFÜHRUNGSFORMEN

### Überblick

**[0020]** Die Fig. 1 stellt einen herkömmlichen IVR-Typ des Systems dar. In einem derartigen System kommuniziert eine Person (nicht gezeigt) mit einem Server-Computer **110**. Der Server-Computer **110** ist mit einem Datenbankspeichersystem **112** verbunden, das den Code und die Daten zum Steuern des Betriebs vom Server-Computer **110** beim Führen eines Dialogs mit dem Anrufer enthält. Der Server-Computer **110** ist wie gezeigt mit einem öffentlichen Telefonnetz (PSTN) **114** verbunden, das dann wiederum einen Zugriff für Anrufer über Telefone wie z. B. das Telefon **116** bereitstellt. Wie erwähnt, werden derartige sprach-aktivierte Systeme in einer breiten Vielzahl von Anwendungen verwendet, welche Voice-Mail, Call-Center, Bankwesen, usw. einschließen.

**[0021]** Früher hätten Sprachanwendungsentwickler eine Spracherkennungsmaschine und einen Code, ein anwendungs-spezifisches und spracherkennungsmaschinen-spezifisches System gewählt, das vom Entwickler gefordert hat, jedes einzelne Detail des Dialogs zu handhaben und das gesamte Universum von möglichen Ereignissen vorauszusehen und bereitzustellen. Derartige Anwendungen mussten für

eine neue Anwendung vollständig neugeschrieben oder eine unterschiedliche Spracherkennungsmaschine musste verwendet werden.

**[0022]** Im Gegensatz zum Stand der Technik und in Bezug auf die Fig. 2 stellt die vorliegende Erfindung ein System bereit, das Entwickler von zeitintensiven, maschinennahe Programmierungsaufgaben gelöst, indem es dem Entwickler ermöglicht, verallgemeinerte Befehle über den Fluss einer Konversation (das potenziell viele Zustände oder Wechsel einer Konversation einschließt) an ein Entwicklungswerkzeug für einen Dialogflussinterpreter (DFI) **210** festzulegen, auf das über eine Programmierer-freundliche grafische Schnittstelle (nicht gezeigt) zugegriffen werden kann. Das DFI-Entwurfswerkzeug **210** erzeugt eine Datendatei **220** (eine Benutzeroberfläche der Anwendung). Wenn das aufrufende Programm (Sprachanwendung) **230**, das vom Entwickler in einer Vielzahl von Programmiersprachen geschrieben sein kann, ausgeführt wird, instantiiert das aufrufende Programm **230** den Dialogflussinterpreter **232**, wobei es dem Interpreter **232** die Datendatei **220** bereitstellt, die vom DFI-Entwurfswerkzeug **210** erzeugt wurde. Der Dialogflussinterpreter **232** ruft dann Funktionen des DFI-Objekts auf, um die Sprachanwendung zu implementieren, wobei er sämtliche Details der Zustandshandhabung und des Konversationsflusses bereitstellt, die zuvor der Programmierer schreiben musste. Nachdem das aufrufende Programm **230** einmal geschrieben wurde, kann es für verschiedene Anwendungen verwendet werden. Die Anwendungen unterscheiden sich von einander im Inhalt der Eingabeaufforderungen und der erwarteten Antworten und in der resultierenden Verarbeitung (Verzweigungen und Konversationsfluss) und in der verwendeten Spracherkennungsmaschine, von denen sämtliche gemäß der vorliegenden Erfindung in der Datendatei **220** gespeichert sein können. Folglich kann das bestehende aufrufende Programm **230** durch Wechseln der Datendatei **220** für unterschiedliche Anwendungen verwendet werden.

**[0023]** Das Entwicklungswerkzeug **200** speichert automatisch den wiederverwendbaren Code von irgendeiner Detailstufe, wobei es Dialogobjekte einschließt, in einer Bibliothek, die für die Verwendung in anderen Anwendungen zugreifbar gemacht werden kann. Ein Dialogobjekt ist eine Sammlung von einem oder mehreren Dialogzuständen, welche die Verarbeitung einschließt, die im Aneinanderhängen der Zustände eingeschlossen ist.

**[0024]** Da die durch das Entwicklungsprogrammierwerkzeug erschaffenen Sprachanwendungen als ausführbare Objekte implementiert werden, können die Anwendungen einfach in eine Vielzahl von verschiedenen Plattformen integriert werden. Eine Anzahl von verschiedenen Spracherkennungsmaschinen kann unterstützt werden. Die spezielle Spracherkennungsmaschine, die in einer speziellen Anwendung verwendet wird, kann einfach gewechselt werden. Wir werden nun die vorliegende Erfindung detaillierter erklären, indem sie mit dem Stand der Technik verglichen wird.

Stand der Technik

**[0025]** Wieder mit Bezug auf die Fig. 1 ist der gewöhnlichste Weg für einen Benutzer in einem Dialog-basierten System mit einem Computer zu kommunizieren über ein Mikrofon oder über ein Telefon **116**, das über ein Fernsprechvermittlungssystem **114** mit einem Computer verbunden ist, auf dem die Software, die es dem Menschen und dem Computer zu interagieren ermöglicht, in einer Datenbank **112** gespeichert ist. Jede Interaktion zwischen dem Computer und dem Benutzer, in welcher der Computer versucht, ein bestimmtes Stück von Information vom Benutzer herauszuholen, wird ein Zustand oder ein Wechsel genannt. In jedem Zustand beginnt der Computer mit einer Eingabeaufforderung und der Benutzer gibt eine gesprochene Antwort. Die Anwendung muss das, was der Benutzer gesagt hat, erkennen und interpretieren, die passende auf dieser Antwort basierende Handlung ausführen und dann die Konversation zu dem nächsten Zustand oder Wechsel führen. Die Schritte sind wie folgt:

1. Der Computer gibt eine Eingabeaufforderung heraus.
2. Der Benutzer (oder der Anrufer) antwortet.
3. Der Spracherkenner wandelt die Antwort in eine Computerlesbare Form um.
4. Die Anwendung interpretiert die Antwort und handelt dementsprechend. Dies kann zum Beispiel einen Datenbankzugriff für eine Abfrage einschließen.
5. Die Anwendung kann dem Benutzer antworten.
6. Die Schritte 1 bis 5 können wiederholt werden bis eine zufriedenstellende Antwort vom Benutzer empfangen wird.
7. Die Anwendung geht in den nächsten Zustand über.

**[0026]** Somit schließt eine Dialog-basierte Sprachanwendung einen Satz von Zuständen ein, die einen Benutzer zu seinem Ziel führen. Zuvor musste der Entwickler jeden Schritt im Dialog codieren, indem er für jedes mögliche Ereignis und jede mögliche Antwort im Universum der möglichen Ereignisse codiert hat, eine zeitintensive und technisch komplexe Aufgabe. Der Entwickler musste ein interaktives Sprachausgabesystem (IVR) auswählen, wie zum Beispiel das Parity, und die Anwendung in der Programmiersprache codieren, die mit jener Sprache verbunden war, wobei eine Spracherkennungsmaschine wie Nuance, Lernout und Hauspie oder eine andere Spracherkennungsmaschine verwendet wurde, die in die IVR-Umgebung einschließbar war.

**[0027]** Sprachobjekte sind kommerziell erhältlich. Mit Bezug auf die Fig. 3 stellen die Sprachobjekte **322**, **324** vorgefertigte Bits von sämtlichen Dingen dar, die in eine Sprachhandlung einfließen, typischerweise eine Eingabeaufforderung, eine Grammatik und eine Antwort. In diesem Schema wird ein Sprachobjekt, zum Beispiel das "Hole die Sozialversicherungsnummer" **322**, von einem Verkäufer bezogen. Ein Entwickler schreibt den Software-Code **320** in der für die ausgewählten Sprachobjekte erforderlichen Programmiersprache, und platziert das bezogene Sprachobjekt "Hole die Sozialversicherungsnummer" **322** in seiner Software. Wenn das Programm ausgeführt wird und einen Punkt erreicht, an dem die Sozialversicherungsnummer benötigt wird, dann wird das Sprachobjekt "Hole die Sozialversicherungsnummer" **322** aufgerufen. Die Anwendung kann sich geringfügig geändert haben, dadurch wie die Frage gestellt wurde, aber der Flexibilitätsbereich des Sprachobjekts ist beschränkt. Nachdem die Antwort vom Benutzer erhalten wurde, wird die Steuerung an die Anwendung **320** zurückgegeben. Die vom Entwickler geschriebene Anwendung **320** muss dann den Übergang zu dem nächsten Zustand, die "Hole die PIN-Nummer" **324** usw., handhaben. Sprachobjekte werden an einem spezifischen Entwicklungssystem (z. B. das "Speech Channels" genannte "IVR-System" von Nuance und das als ein integriertes Anwendungssoftwarepaket bezeichnete "IVR-System" von SpeechWorks) implementiert. Diese wiederverwendbaren Teile sind nur innerhalb der Umgebung wiederverwendbar, für die sie erstellt wurden. Zum Beispiel wird eine SpeechWorks-Implementierung hiervon, die "Dialog Modules" genannt wird, nur innerhalb dem integrierten SpeechWorks-Anwendungssoftwarepaket funktionieren. Die Kernlogik ist nicht wiederverwendbar, weil es an die Implementierungsplattform gebunden ist.

DFI-Entwurfswerkzeug

**[0028]** Im Gegensatz, gemäß der vorliegenden Erfindung und in Bezug auf die Fig. 4 benutzt der Entwickler das DFI-Entwurfswerkzeug **400**, um einen Entwurf der gesamten Anwendung einzugeben, wie im Schritt **410** dargestellt ist, der viele derartige Zustände wie "Hole die Sozialversicherungsnummer", "Hole die PIN-Nummer" und so weiter einschließt. Sobald die Anwendung im Simulator erprobt worden ist, Schritt **420** (siehe als Referenz das Patent mit der Seriennummer 09/417.166), können Dateien erzeugt werden, die diesen Entwurf darstellen, Schritte **440** und **450**.

**[0029]** Wie in der Fig. 5 gezeigt wird, instantiiert die Softwareanwendung **510**, die vom Entwickler in irgendeiner der Vielzahl von Programmiersprachen codiert wurde, den Dialogflussinterpreter **530** und sagt ihm, den Entwurf zu interpretieren, der in der Datei **520** festgelegt ist, die oben vom DFI-Entwurfswerkzeug erzeugt wurde. Der Dialogflussinterpreter **530** steuert den Fluss durch die Anwendung, die den gesamten zugrundeliegenden Code **540** liefert, den zuvor der Entwickler schreiben musste.

**[0030]** Wie aus der **Fig.** 6A, 612 und der **Fig.** 6B, 622 ersichtlich ist, wird die Menge an Code, die von einem Programmierer geschrieben werden muss, erheblich verringert. Tatsächlich kann sie in einigen Anwendungen vollkommen eliminiert werden.

Dialogflussinterpreter

**[0031]** Der Dialogflussinterpreter oder der DFI der vorliegenden Erfindung stellt eine Bibliothek von "standardisierten" Objekten bereit, die maschinennahe Details von Dialogen implementieren. Der DFI kann als eine Anwendungsprogramm-Schnittstelle (API) implementiert werden, was die Implementierung von Sprachanwendungen vereinfacht. Die Sprachanwendungen können entworfen sein, indem ein Werkzeug verwendet wird, das als das DFI-Entwicklungswerkzeug bezeichnet wird. Die von der Erfindung bereitgestellte Vereinfachung rührt von der Tatsache, dass der DFI in der Lage ist, den gesamten Dialog einer Sprachanwendung vom Anfang bis zum Ende automatisch zu führen, und somit die entscheidende und oftmals komplexe Aufgabe der Dialogverwaltung eliminiert. Herkömmlich ist ein derartiger Vorgang anwendungsabhängig und erfordert deswegen eine Neuimplementierung für jede neue Anwendung. Der DFI löst dieses Problem, indem es eine "einmal Schreiben"-"oft ausgeführt"-Lösung bereitstellt.

**[0032]** Die Fig. 2 stellt die Beziehung zwischen dem DFI-Entwurfswerkzeug **210**, dem Dialogflussinterpreter **232** und anderen Sprachanwendungskomponenten dar. (In diesem Diagramm stellen Block-pfeile die Richtung des Datenflusses dar.)

Funktionselemente

**[0033]** Eine Sprachanwendung schließt eine Reihe von Übergängen zwischen Zuständen ein. Jeder Zustand besitzt seinen eigenen Satz von Eigenschaften, der die abzuspielende Eingabeaufforderung, die zu ladende Grammatik des Spracherkenners (nach dem zu horchen, was der Benutzer des Sprachsystems sagen könnte), die Erwiderung auf die Antwort eines Anrufers und Handlungen einschließt, die basierend auf jeder Antwort auszuführen sind. Der DFI verfolgt den Zustand des Dialogs zu jedem gegebenen Zeitpunkt während dem Bestehen der Anwendung und bietet Funktionen, um auf Zustandseigenschaften zuzugreifen.

**[0034]** Mit Bezug auf die Fig. 7 ist ersichtlich, dass Zustandseigenschaften in Objekten gespeichert werden, die als "gemeinsam benutzte Objekte" (shared objects) **710** bezeichnet werden. Beispiele dieser

Objekte beinhalten, sind aber nicht darauf beschränkt, ein Eingabeaufforderungsobjekt, ein Schnipselobjekt (snippet object), ein Grammatikobjekt, ein Antwortobjekt, ein Handlungsobjekt und ein Variableobjekt.

**[0035]** Beispielhaft geben die DFI-Funktionen **780** einige der oben beschriebenen Objekte zurück. Diese Funktionen schließen folgendes ein:

**[0036]** "Hole die Eingabeaufforderung" **720**: Gibt die passende abzuspielende Eingabeaufforderung zurück. Diese Eingabeaufforderung wird dann an die passende Tonabspielroutine für die Tonausgabe weitergegeben.

**[0037]** "Hole die Grammatik" **730**: Gibt die passende Grammatik für den gegenwärtigen Zustand zurück. Diese Grammatik wird dann in die Spracherkennungsmaschine geladen.

**[0038]** "Hole die Antwort" **740**: Gibt ein Antwortobjekt zurück, das die aktuelle Benutzerantwort, irgendeine der Abweichungen, die diese Antwort enthalten kann, und sämtliche möglichen Handlungen, die für diese Antwort definiert sind, umfasst.

**[0039]** "Schreite den Zustand fort" **750**: Führt den Dialog in den nächsten Zustand über.

**[0040]** Andere DFI-Funktionen werden verwendet, um Zustand-unabhängige Eigenschaften zu erlangen (d. h. globale Projekteigenschaften). Diese beinhalten, sind aber nicht darauf beschränkt:
den Projektpfad **760**
den Pfad der Projekttöne
die Eingabeart (DTMF oder Ton)
die Aufschaltungsart (DTMF oder Ton)
den gegenwärtiger Zustand
den vorherigen Zustand

Alternative DFI-Verwendungen

**[0041]** Das Registriergerät für die Dialog-Metrik – Weil der DFI das Interne des Übergangs zwischen den Zuständen steuert, wäre es eine einfache Angelegenheit zu zählen, wie oft ein bestimmter Zustand eingenommen wurde, zum Beispiel so, dass Statistiken darüber gesammelt werden könnten, wie eine Sprachanwendung verwendet wird oder wie eine Sprachanwendung arbeitet.

**[0042]** Belastungstester für die Sprachanwendung – Weil der DFI das Interne des Übergangs zwischen den Zuständen steuert, ermöglicht das DFI-Werkzeug die Entwicklung einer Anwendung (bei der Verwendung von Text zu Sprache), die das Testen von Sprachanwendungen erleichtern würde, indem sie die menschliche Seite des Dialogs zusätzlich zur Computerseite des Dialogs vorsieht.

**[0043]** Die Fig. 7 stellt dar, wie die DFI-Funktionen **780** als eine Anwendungsprogramm-Schnittstelle (API) implementiert oder betrachtet werden können.

Vergleich des DFI mit Sprachobjekten

**[0044]** Sprachobjekte (ein gewöhnliches Konzept in der Industrie) stellen vorgefertigte Bits von sämtlichen Dingen dar, die in eine "Sprachhandlung" einfließen, typischerweise eine Eingabeaufforderung (etwas zu sagen), eine Grammatik (nach etwas zu horchen) und vielleicht irgendeine Art von Reaktion auf der Seite des Systems. Das könnte das Zusammentragen eines einzelnen Informationsbits abdecken (was einfach erscheint solange man alles, was schief laufen könnte, berücksichtigt). Ein Ansatz besteht darin, eine vorgefertigte Funktionalität anzubieten (z. B. SpeechWorks (www.speechworks.com)). Ein Beispiel des grundlegenden Modells ist wie folgt: Der Entwickler kauft (z. B. von Nuance) ein Sprachobjekt, das "Hole die Sozialversicherungsnummer" genannt wird, und setzt es in sein Programm ein. Wenn das Programm einen Punkt erreicht, an dem die Sozialversicherungsnummer eines Benutzers benötigt wird, ruft der Entwickler das "Hole die Sozialversicherungsnummer"-Objekt auf. Die Anwendung könnte es ein wenig abgeändert haben, indem sie genau verändert, wie die Frage gestellt wird oder die Breite dessen ausweitet, was sie hören wird, aber der Grundwert ist die vorgefertigte Methodologie und die voreingestellte Funktionalität des Objekts.

**[0045]** Im Entwicklungswerkzeug für einen Dialogflussinterpreter der vorliegenden Erfindung benutzt der Entwickler ein Entwurfswerkzeug (sprich das von der Unisys Corporation angebotene DFI-Werkzeug), um einen Entwurf der gesamten Anwendung einzugeben (die potenziell viele Zustände wie "Holen der SV-Nr." und "Holen. der PIN" und so weiter einschließt). Sobald diese Anwendung in einem Simulator erprobt worden ist ("Wizard of Oz"-Tester), werden Dateien erzeugt, die diesen Entwurf darstellen (z. B. MySpeechApp). Der DFI wird von der "Laufzeit"-Anwendung instantiiert (die in irgendeiner Programmiersprache geschrieben wurde) und angewiesen, den Entwurf zu interpretieren (MySpeech-App), der vom Entwurfswerkzeug erstellt wurde. Sobald die Einrichtung erfolgt ist, braucht der Anwendungscode dem DFI nur die Details dessen zu geben, was vor sich geht, um den Entwurf für das "zurückzulesen", was als nächstes zu tun ist. So kann zum Beispiel der Entwerfer eine Sequenz wie folgt angeben:
Wie ist Ihre SV-Nummer?
(nach der SV-Nummer horchen)
Wie lautet Ihre PIN?
(nach der PIN horchen)
Wollen Sie anweisen oder ein Problem melden?
(nach "Anweisen" oder "Ein Problem melden" horchen)
wenn "Anweisen", dann

Wie ist Ihre Anweisung ...
sonst wenn "Ein Problem melden", dann
Was ist Ihr Problem ...

**[0046]** In diesem Fall nimmt der DFI zuerst einen Zustand ein, in dem, als das Programm gefragt hatte welche Eingabeaufforderung abzuspielen sei, es "Wie ist Ihre SV-Nummer?" antwortet und angibt, dass das Programm nach der SV-Nr. horchen soll. Sobald die Anwendung dem DFI mitgeteilt hat, dass dies ausgeführt worden ist und fortzufahren sei, fragt die Anwendung wieder den DFI, was zu sagen sei und es antwortet nun "Wie ist Ihre PIN". Der DFI fährt solange fort, direktionale Daten zu liefern, bis die Anwendung geendet hat. Der Punkt ist der, dass der DFI das "Interne" für jeden Wechsel des Dialogs (Eingabeaufforderung, nach was zu horchen, usw.) als auch den Fluss durch die Anwendung liefert.

**[0047]** Obwohl sie ähnliche Probleme ansprechen, ist der DFI sehr verschieden vom Modell der Sprachobjekte. Sprachobjekte richten Grundeinstellungen ein, über die sich ein Programm hinwegsetzen kann (das Programm muss das von irgendwo wissen), wohingegen der DFI die Anwendung damit versorgt, was als nächstes zu tun ist. Sprachobjekte sind starr und vorprogrammiert und von beschränktem Umfang, wohingegen der DFI für eine ganze Anwendung erstellt und dynamisch ist. Sprachobjekte sind für einen besonderen Zweck "eingestellt", nämlich dem zuvor genannten. Diese Einstellung kann auch über das DFI-Entwurfswerkzeug bereitgestellt werden. Eine andere Möglichkeit, um sich den Unterschied bewusst zu machen besteht darin, dass der DFI über das Werkzeug erstellte "kundenspezifische" Sprachfähigkeiten liefert, die das, wie sie aneinander "hängen" einschließt. Sprachobjekte stellen "vorgefertigte" Fähigkeiten (mit dem Vorteil des "Fachmannentwurfs" und der Einstellung) und ohne einen "Fluss" zwischen ihnen bereit.

Übersetzerobjektklassen

**[0048]** Eine Sprachanwendung muss in der Lage sein, Information in einer Form zu erlangen, welche die Software interpretieren kann. Sobald die Information erhalten wurde, kann es wünschenswert sein, diese Information in einem speziellen Sprachformat an die Außenwelt auszugeben. Gemäß der vorliegenden Erfindung ermöglichen es Übersetzerobjektklassen einem Entwickler, Parameter bereitzustellen, um Details darüber festzulegen, wie ein spezielles Stück an Information ausgeben werden sollte, und der DFI wird alles Notwendige zurückgeben, um diese Aufgabe auszuführen. Wenn zum Beispiel das gewünschte Objekt in Englisch in der Standardzeit auszugeben ist, wie spät es gegenwärtig in Belgien ist, legt der Entwickler die Sprache (Englisch), das Gebiet (Belgien), die Zeit (die momentane Zeit in Belgien) und das Format (die Standardzeit) fest, und der DFI gibt eine Programmliste von allem Erforderlichen zurück, um es dem Zuhörer zu ermöglichen, die Datenstruktur mit jenen Kennzeichen (die momentane Zeit in Belgien im Standardformat, auf Englisch gesprochen) zu hören.

**[0049]** Wenn zum Beispiel der DFI das Eingabeauffordern beendet hat, greift der DFI auf die Funktion "Hole die Eingabeaufforderung" **720** zu, Fig. 7, die zurückgegeben worden wäre (wenn die ausgegebene Sprache eine aufgezeichnete Datei ist):

1. die Datei "Es ist jetzt".wav,
2. den Wert der Zeitinstanz (variabel), 12:35 nachmittags, und die zugehörigen Dateien:
   zwölf.wav
   dreißig.wav
   fünf.wav
   nachmittags.wav,
3. und die Datei "in Belgien".wav.

**[0050]** Der Zuhörer würde hören: "Es ist jetzt zwölf-fünfunddreißig nachmittags in Belgien". Es sollte verstanden werden, dass das obige Beispiel nur für exemplarische Zwecke ist. Die vorliegende Erfindung einschließt auch die "Text zu Sprache" (Computererzeugte) Sprachausgabe.

**[0051]** Ersatzweise, wenn der Entwickler das Objekt in seiner Anwendung unmittelbar verwenden will, ohne den DFI zu verwenden, kann die Anwendung auf den Übersetzer unmittelbar zugreifen. Der Übersetzer sendet den Wert der Zeitinstanz (12:35) und die zugehörigen Dateien zurück:
zwölf.wav
dreißig.wav
fünf.wav
nachmittags.wav

**[0052]** Somit enthalten die Übersetzerobjektklassen Objekte, die von der Sprachanwendung verwendet werden können, die vom Entwickler oder vom DFI geschrieben wurde. Obwohl kommerziell erhältliche Sprachobjekte eine ähnliche Funktionalität bereitstellen können, liegt das Erfinderische der Übersetzerobjektklassen darin, dass der Entwickler die Steuerung der maschinennahen Details nicht verliert; wie die Information ausgegeben wird, weil der Entwickler seine eigenen zu der Klasse beizufügenden Objekte schreiben kann. Wenn ein Entwickler kommerziell erhältliche Sprachobjekte verwendet, muss der Entwickler den Flexibilitätsverlust in Kauf nehmen, um die Arbeitsweise des Sprachobjekts zu steuern. Mit Übersetzerobjekten gemäß der vorliegenden Erfindung hält der Entwickler die Steuerung der maschinennahen Details aufrecht, während die maximale Menge an Automation stets erlangt wird.

Schlussfolgerung

**[0053]** Zusammenfassend stellt die vorliegende Er-

findung ein System und Verfahren bereit, um interaktive Dialoge zwischen einem Menschen und einem Computer zu erzeugen, wie z. B. in einem IVR-System oder dergleichen. Es wird jedoch verstanden, dass die Erfindung verschiedene Veränderungen und alternative Konstruktionen zulässt. Es besteht keine Absicht, die Erfindung auf die hierin beschriebenen spezifischen Konstruktionen zu beschränken. Die Erfindung beabsichtigt im Gegenteil, sämtliche Veränderungen, alternative Konstruktionen und Äquivalente abzudecken, die in den Schutzumfang der Erfindung fallen. Die vorliegende Erfindung kann zum Beispiel nicht sprach-aktivierte Anwendungen unterstützen, in denen ein Computer und ein Mensch interagieren. Die vorliegende Erfindung gestattet den Wiederaufruf der Textbeschreibung einer Eingabeaufforderung, die schriftbildlich dar gestellt werden kann, wobei der Benutzer durch Eintippen in ein Bearbeitungsfeld antwortet. Mit anderen Worten sind es der Dialogfluss und die Eigenschaften von jedem Zustand, was den Kern der Erfindung darstellt und nicht die Verwirklichung des Dialogs. Eine derartige Ausführungsform kann in einem Computerspiel oder innerhalb einer Software verwendet werden, die eine Konfigurationsinformation sammelt, oder in einer Internetanwendung, die interaktiver ist als das, was einfache grafische Benutzerschnittstellentechniken (GUI) ermöglichen.

**[0054]** Es soll auch angemerkt werden, dass die vorliegende Erfindung in einer Vielzahl von Computerumgebungen implementiert werden kann. Zum Beispiel kann die vorliegende Erfindung in Java implementiert werden, wobei sie einen direkten Zugriff aus irgendeiner Java-Programmiersprache ermöglicht. Zusätzlich kann die Implementierung von einer COM-Schicht umhüllt sein, die es irgendeiner Sprache, die das COM unterstützt, auf die Funktionen zuzugreifen gestattet, und es somit traditionellen Entwicklungsumgebungen wie Visual Basic, C/C++, usw. ermöglicht, die vorliegende Erfindung zu verwenden. Auf die vorliegende Erfindung kann auch ausgehend von Microsoft-Anwendungen zugegriffen werden, die von Word, Excel, usw. bis zum Beispiel das Visual Basic für Anwendungen (VBA) beinhalten, aber nicht darauf beschränkt sind. Herkömmliche DTMF-orientierte Systeme, wie zum Beispiel das Parity, die kommerziell erhältlich sind, können die vorliegende Erfindung in ihre Plattform einbetten. Die vorliegende Erfindung und dessen zugehörige Objekte können auch in Entwicklungsumgebungen für das World Wide Web und das Internet eingesetzt werden, wobei es der Hypertext Markup Language (HTML) und ähnlichen Protokollen ermöglicht wird, auf das DFI-Entwicklungswerkzeug und auf dessen Objekte zuzugreifen.

**[0055]** Die hierin beschriebenen unterschiedlichen Techniken können in eine Hardware oder eine Software oder in eine Kombination von beidem implementiert werden. Vorzugsweise werden die Techniken in Computerprogramme implementiert, die auf programmierbaren Computern ablaufen, von denen jeder einen Prozessor, ein vom Prozessor lesbares Speichermedium (das flüchtige und nichtflüchtige Speicher und/oder Speicherelemente einschließt), wenigstens ein Eingabegerät und wenigstens ein Ausgabegerät einschließt. Der Programmcode wird auf eingegebene Daten angewendet, wobei das Eingabegerät verwendet wird, um die oben beschriebenen Funktionen auszuführen und die Ausgabeinformation zu erzeugen. Die Ausgabeinformation wird auf ein oder mehrere Ausgabegeräte angewendet. Jedes Programm wird vorzugsweise in eine höhere verfahrensorientierte oder objektorientierte Programmiersprache implementiert, um mit einem Computersystem zu kommunizieren. Die Programme können jedoch in die Assembler- oder Maschinensprache implementiert werden, wenn es erwünscht wird. In jedem Fall kann die Sprache eine kompilierte oder interpretierte Sprache sein. Jedes derartige Computerprogramm wird vorzugsweise auf einem Speichermedium oder -gerät (z. B. ein ROM oder eine Diskette) gespeichert, die von einem für einen allgemeinen oder speziellen Zweck programmierbaren Computer zum Konfigurieren und Betreiben des Computers gelesen werden kann, wenn das Speichermedium oder -gerät vom Computer gelesen wird, um die oben beschriebenen Abläufe auszuführen. Es kann auch in Erwägung gezogen werden, das System als ein Computer-lesbares Speichermedium zu implementieren, das mit einem Computerprogramm konfiguriert ist, wo das derart konfigurierte Speichermedium einen Computer dazu veranlasst, in einer spezifischen und vordefinierten Weise zu arbeiten.

**[0056]** Obwohl oben eine exemplarische Implementierung der Erfindung im Detail beschrieben wurde, wird der Fachmann leicht verstehen, dass in den beispielhaften Ausführungsformen viele zusätzliche Veränderungen möglich sind, ohne wesentlich von den neuen Lehren und Vorteilen der Erfindung abzuweichen. Dementsprechend ist es beabsichtigt, diese und sämtliche derartige Veränderungen innerhalb des Schutzumfangs dieser Erfindung wie in den angefügten Ansprüchen definiert wird zu umfassen.

## Patentansprüche

1. Ein Verfahren zur Entwicklung einer Dialog-aktivierten Anwendung zum Ausführen auf einem Computer, die es einem Menschen und einem Computer ermöglicht, zu interagieren, das die folgenden Schritte umfasst:
(a) Eingeben von Befehlen, die den Fluss einer Konversation zu einem Entwurfswerkzeug festlegen, wobei das Entwurfswerkzeug eine Datendatei erzeugt und wobei die Datendatei Information über Eingabeaufforderungen, Antworten, Verzweigungen und Konversationsfluss zum Implementieren einer

sprach-aktivierten Mensch-Computer-Interaktion enthält; und
(b) Instantiieren eines Interpreterobjekts innerhalb der Anwendung, wobei das Interpreterobjekt die Datendatei zusammen mit einer Bibliothek von gemeinsam benutzten Objekten interpretiert, um die Dialog-aktivierte Mensch-Computer-Interaktion bereitzustellen, die von der Datendatei definiert wird.

2. Das Verfahren gemäß Anspruch 1, wobei die Datendatei ferner Information über einen Spracherkennungsmechanismus enthält.

3. Das Verfahren gemäß Anspruch 1, wobei die Datendatei automatisch gespeichert wird.

4. Das Verfahren gemäß Anspruch 1, wobei die Eingabe der Befehle über eine grafische Schnittstelle erfolgt.

5. Ein System zum Entwickeln einer Dialog-aktivierten Software zum Ausführen auf einem Computer, die es einem Menschen und einem Computer ermöglicht, zu interagieren, das folgendes umfasst:
ein Entwurfswerkzeug zum Annehmen von Befehlen, die den Fluss einer Konversation festlegen, wobei das Entwurfswerkzeug eine Datendatei erzeugt und wobei die Datendatei Information über Eingabeaufforderungen, Antworten, Verzweigungen und Konversationsfluss zum Implementieren einer sprach-aktivierten Mensch-Computer-Interaktion enthält; und
einen Interpreter zum Interpretieren der Datendatei zusammen mit einer Bibliothek von gemeinsam benutzten Objekten, wobei der Interpreter die Mensch-Computer-Interaktion automatisch ermöglicht.

6. Das System gemäß Anspruch 5 ferner eine Bibliothek umfassend, wobei die Bibliothek die Datendateien enthält.

7. Das System gemäß Anspruch 5, wobei das Entwurfswerkzeug ferner eine "grafische Schnittstelle umfasst.

8. Ein Computer-lesbares Medium, das Computer-ausführbare Anweisungen zum Anweisen eines Computers umfasst, um folgende Handlungen auszuführen:
Annehmen von Befehlen, wobei die Anweisungen einen Fluss von Konversation zwischen einem Menschen und einem Computer festlegen;
Erzeugen einer Datendatei zur Eingabe an einen Interpreter, wobei die Datendatei Information über Eingabeaufforderungen, Antworten, Verzweigungen und Konversationsfluss zum Implementieren einer sprach-aktivierten Mensch-Computer-Interaktion enthält;
Interpretieren der Datendatei zusammen mit einer Bibliothek von gemeinsam benutzten Objekten; und
Bereitstellen der Dialog-aktivierten Mensch-Computer-Interaktion, die von der Datendatei definiert wird.

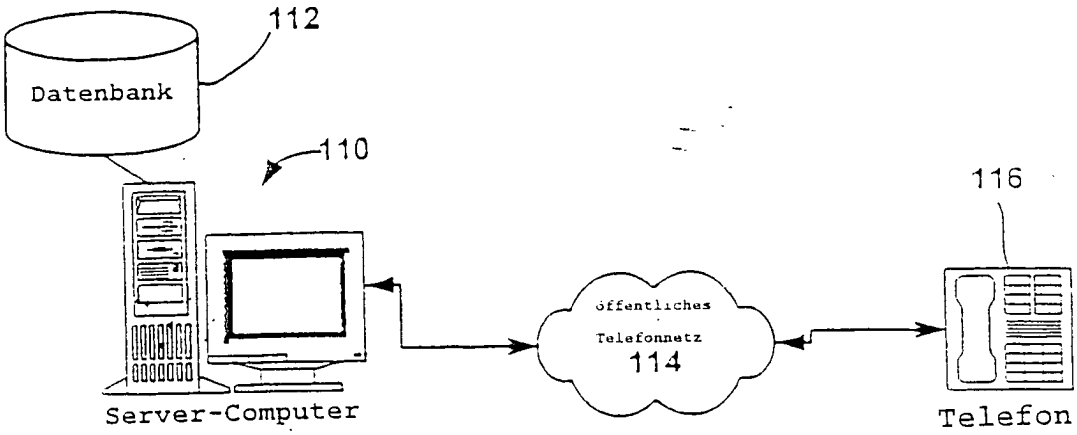
9. Das Computer-lesbare Medium gemäß Anspruch 8, das ferner Befehle enthält, die es ermöglichen, dass auf den erzeugten Code durch andere Softwareentwickler sofort zugegriffen werden kann.

10. Ein Dialogflussinterpreter (DFI) zur Verwendung in einem Computer-implementierten System zum Ausführen eines Dialogs zwischen einem Menschen und einem Computer, wobei der DFI Computer-ausführbare Befehle zum Lesen einer Datendatei umfasst, die Information über Eingabeaufforderungen, Antworten, Verzweigungen und Konversationsfluss zum Implementieren eines sprach-aktivierten Mensch-Computer-Dialogs enthält, und einen Computerausführbaren Code zum Verwenden der Information zusammen mit einer Bibliothek von gemeinsam benutzten Objekten, um den Dialog zu führen.

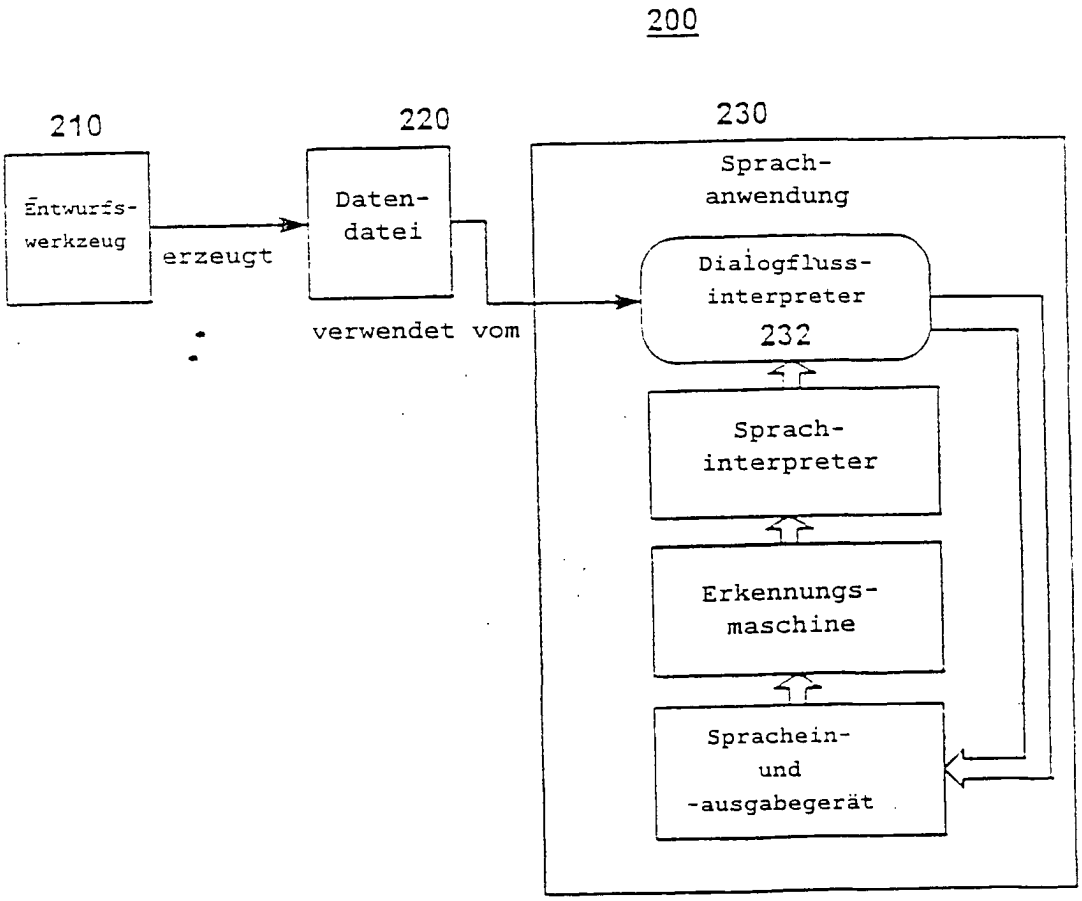
11. Ein DFI gemäß Anspruch 10, wobei der DFI in einer Anwendung implementiert ist, die zusätzlich zu dem DFI einen Sprachinterpreter, ein Erkennungsmechanismus und ein Sprachein- und -ausgabegerät umfasst.

Es folgen 4 Blatt Zeichnungen

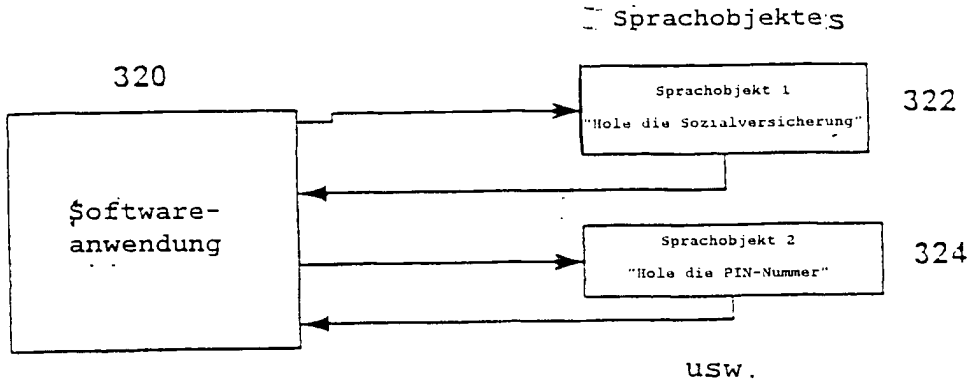
**Anhängende Zeichnungen**



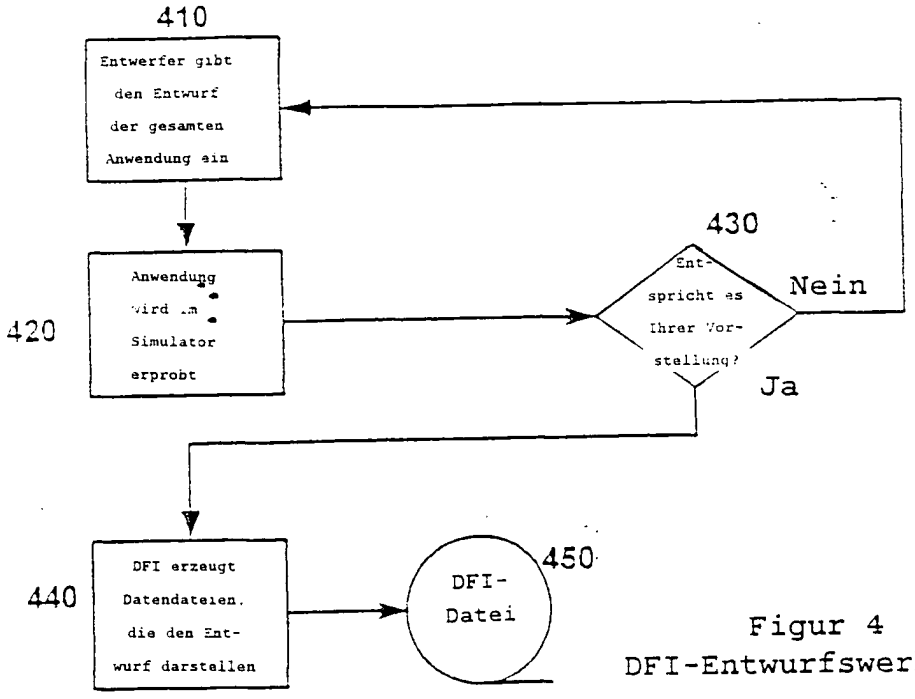

Figur 1 (Stand der Technik)



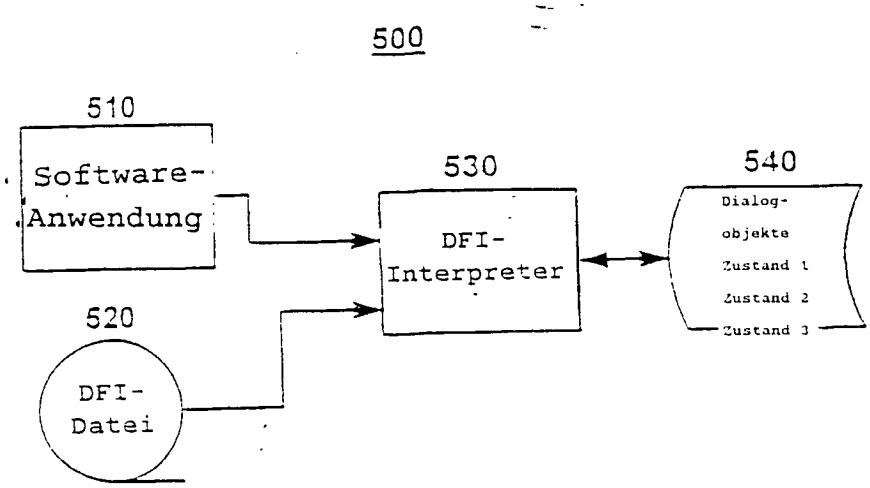

Figur 2
DFI-Entwicklungswerkzeug



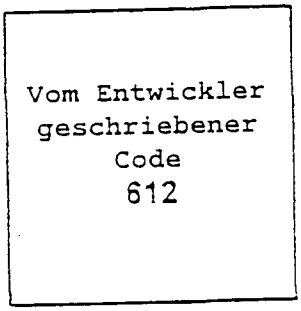

Figur 3 - Stand der Technik



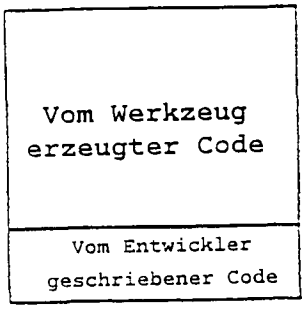

Figur 4
DFI-Entwurfswerkzeug



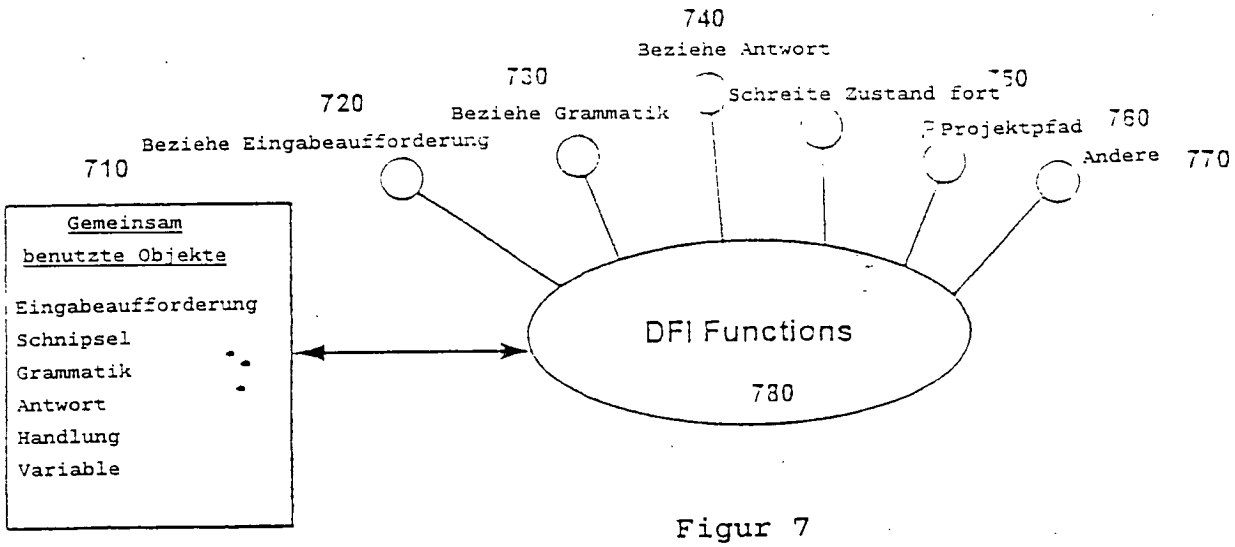

Figur 5
DFI




Figur 6(A) - Stand der Technik




Figur 6(B)
DFI-Werkzeug




Figur 7
DFI-Funktionen